Assignment 2: Coding Basics

Yared S. Asfaw

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on coding basics.

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your first and last name into the file name (e.g., "FirstLast_A02_CodingBasics.Rmd") prior to submission.

Basics Day 1

1. Generate a sequence of numbers from one to 100, increasing by fours. Assign this sequence a name.

2. Compute the mean and median of this sequence.

3. Ask R to determine whether the mean is greater than the median.

4. Insert comments in your code to describe what you are doing.

```r
#1. Sequencing numbers from 1 to 100 counting by 4
number_seq <- seq(1, 100, 4)
print(number_seq) # Sequence of numbers from 1 to 100 counted by 4
```

```
##  [1]  1  5  9 13 17 21 25 29 33 37 41 45 49 53 57 61 65 69 73 77 81 85 89 93 97
```

```r
#2. Mean and median of the sequenced numbers
number_seq_mean <- mean(number_seq)
number_seq_mean # The mean of the sequenced numbers
```

```
## [1] 49
```

```r
number_seq_median <- median(number_seq)
number_seq_median # The median of the sequenced numbers
```

```
## [1] 49
```

```
#3. Comparing the mean and median
number_seq_mean > number_seq_median # Is the mean greater than the median?
```

```
## [1] FALSE
```

Basics Day 2

5. Create a series of vectors, each with four components, consisting of (a) names of students, (b) test scores out of a total 100 points, and (c) whether or not they have passed the test (TRUE or FALSE) with a passing grade of 50.

6. Label each vector with a comment on what type of vector it is.

7. Combine each of the vectors into a data frame. Assign the data frame an informative name.

8. Label the columns of your data frame with informative titles.

```
#5. Creating series of vectors
# Vectors of four imaginary students names
stu_name <- c("Abi", "Rachel", "Malden", "Sam")
stu_name # Imaginary students names
```

```
## [1] "Abi"    "Rachel" "Malden" "Sam"
```

```
# Vectors of the imaginary students test scores
test_score <- c(48,75, 80, 45)
test_score # Students imaginary test scores
```

```
## [1] 48 75 80 45
```

```
# Vectors of logical values if the student pass or fail
grade <- c(FALSE, TRUE, TRUE, FALSE)
grade # True if the score is greater than 50 or False
```

```
## [1] FALSE  TRUE  TRUE FALSE
```

```
#6.Defining the type of vector created above
typeof(stu_name) # type of the vector stu_name
```

```
## [1] "character"
```

```
# type of the vector test_score
typeof(test_score) # one of the numeric data type
```

```
## [1] "double"
```

```
typeof(grade) # type of the vector grade
```

```
## [1] "logical"
```

```
#7.Combining the vectors into a data frame
student.df <- data.frame(stu_name, test_score, grade)
student.df
```

```
##   stu_name test_score grade
## 1      Abi         48 FALSE
## 2   Rachel         75  TRUE
## 3   Malden         80  TRUE
## 4      Sam         45 FALSE
```

```
#8. Labling the columns of the data frame
colnames(student.df)
```

```
## [1] "stu_name"   "test_score" "grade"
```

```
#Renaming columns
colnames(student.df) <- c("Studnet Name", "Test Score out of 100", "Passed")
colnames(student.df)
```

```
## [1] "Studnet Name"          "Test Score out of 100" "Passed"
```

```
student.df
```

```
##   Studnet Name Test Score out of 100 Passed
## 1          Abi                    48  FALSE
## 2       Rachel                    75   TRUE
## 3       Malden                    80   TRUE
## 4          Sam                    45  FALSE
```

9. QUESTION: How is this data frame different from a matrix?

   Answer: This data frame enable us to store different data types (character, numeric and logical) unlike matrix which allows to store only similar data types.

10. Create a function with an if/else statement. Your function should determine whether a test score is a passing grade of 50 or above (TRUE or FALSE). You will need to choose either the `if` and `else` statements or the `ifelse` statement. Hint: Use `print`, not `return`. The name of your function should be informative.

11. Apply your function to the vector with test scores that you created in number 5.

```
# creating function using the if...else statement
grading <- function(test_score){
  if(test_score >= 50)
    {
    print(TRUE)
    }
  else
    {
      print(FALSE)
    }
  }
```

```r
grading(70) # Checking if a student who scored 70 passed
```

```
## [1] TRUE
```

```r
grading(45) # checking if a student who scored 45 passed
```

```
## [1] FALSE
```

```r
#Another option: creating function using the ifelse statement
grading2 <- function(test_score){
ifelse(test_score>=50, TRUE, FALSE)
 }
```

```r
grading2(30) # checking if a student who scored 30 passed
```

```
## [1] FALSE
```

```r
grading2(90) # checking if a student who scored 90 passed
```

```
## [1] TRUE
```

12. QUESTION: Which option of `if` and `else` vs. `ifelse` worked? Why?

    Answer: Both worked for checking if a test score is 50 or above. However, the ifelse took only one line of code and easy to respond to the question. Whereas the if…else though worked, it took more than one line of code and has many syntax as compared to the ifelse which makes it more complicated than the ifelse